# *FPM: Fine Pose Parts-Based Model with 3D CAD Models*

# FPM: Fine pose Parts-based Model with 3D CAD models

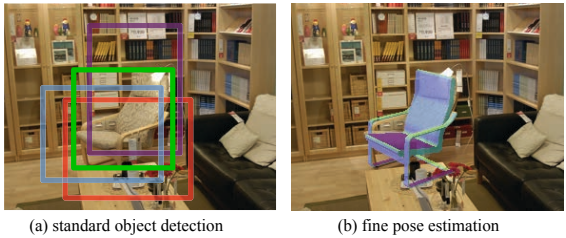
Joseph J. Lim, Aditya Khosla, and Antonio Torralba

Massachusetts Institute of Technology
{lim,khosla,torralba}@csail.mit.edu

**Abstract.** We introduce a novel approach to the problem of localizing objects in an image and estimating their fine-pose. Given exact CAD models, and a few real training images with aligned models, we propose to leverage the geometric information from CAD models and appearance information from real images to learn a model that can accurately estimate fine pose in real images. Specifically, we propose FPM, a fine pose parts-based model, that combines geometric information in the form of shared 3D parts in deformable part based models, and appearance information in the form of objectness to achieve both fast and accurate fine pose estimation. Our method significantly outperforms current state-of-the-art algorithms in both accuracy and speed.

## 1 Introduction

Imagine an autonomous agent attempting to find a chair to sit on. Using a state-of-the-art object detection system, the agent finds a number of *correct* detections (as defined by the 0.5 intersection-over-union criterion) of chairs as shown in Figure 1(a). With the given set of *correct* detections, the agent could end up sitting anywhere from the bookshelf, to the floor! In order to better understand and interact with our environment, it is important to tackle the problem of fine pose estimation as shown in Figure 1(b).



(a) standard object detection (b) fine pose estimation

**Fig. 1.** If an autonomous agent attempted to sit on a chair based on *correct* detections by an object detector (a), who knows where it might end up? Fine pose estimation, shown in (b), is one possible method to tackle this problem.

While fine pose estimation for simple 3D shapes has been studied since the early days of computer vision [28], estimating the fine pose of articulate 3D objects has received little attention. Given the recent success of the computer vision

community in addressing object detection [8, 13] and 3D scene understanding [12, 24, 4, 33, 16], fine pose estimation is becoming more and more approachable. In this work, we tackle the problem of instance-level fine pose estimation given only a single-view image. Specifically, we want to estimate the fine pose of objects in images, given a set of exact 3D models. We restrict our attention to instance-level detection as it provides several advantages over category-level detection. For example, we can get pixel-perfect alignment with per-instance exact 3D models. This would not be possible if we instead used a single generic model for a category, as categories contain large variation in shape across instances.

Recently, there has been an increasing interest in utilizing 3D models for object detection and pose estimation [39, 27, 10, 19, 34]. Despite this, using explicit 3D models for fine pose estimation presents various challenges. First, the space of possible poses of an object is extremely large. Even if we assume that the intrinsic parameters of the camera are known, we still have to estimate at least 6 parameters (translation and rotation) to determine the exact pose of the model. This is *three* more parameters than the typical 2D object detection problem ($x$, $y$, and *scale*), which already has a search space of millions of possible bounding boxes. Second, there is the issue of domain adaptation. While rendered images of 3D models look similar to real objects, there are some key differences that make it difficult to use the same set of features as we do for real images. For example, rendered images are often texture-less and do not contain occluding objects or surfaces.

In this paper, we propose an algorithm that combines appearance information from images with geometric information from 3D models for efficient fine pose estimation. Specifically, we introduce the notion of 3D part sharing that is enabled through the use of 3D models. As illustrated in Figure 2, 2D parts [8, 9] are significantly affected by viewpoint, while 3D parts are shared across views. For each 3D part, we learn an *importance score* that measures the visibility and discriminative-ness of the given part. Our algorithm outperforms existing state-of-the-art methods in both speed and accuracy.

## 2 Related Work

Various recent works have tackled the problem of object-level pose estimation [37, 21, 11]. These studies make assumptions that a major structure is shared between objects, e.g. cuboid, and hence can use this set of relatively few models to infer the pose of objects. In this work, our goal is to differentiate fine details even within the same object category, e.g. our dataset contains at least 3 different types of bookcases. Using this fine differentiation, we can achieve finer pose estimation. There are a few recent papers [27, 36, 29] that have created a dataset for real images annotated with full 3D poses.

There have also been several works [15, 18, 23, 22, 32, 38] addressing the problem of estimating the structure of a scene from a single image. The goal of these works is to recover a scene reconstruction, usually based on vanishing point estimations. These works are complementary to our work in that having recon-

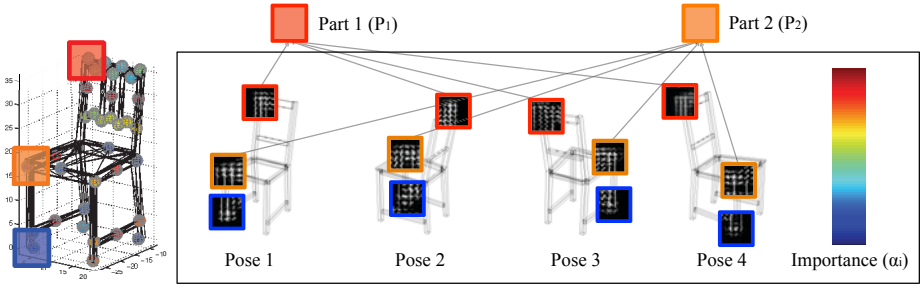



**Fig. 2.** Our goal is to learn a model that can estimate the fine pose of objects from a single image. Like deformable part models [8], we approach this problem by training root and part templates for a large number of views. While each part $P_i$ has its own shape template $w_i$ trained from rendered images, its importance weight $\alpha_i$ is trained with a small set of annotated real images. The goals of importance weight $\alpha_i$ for each part $P_i$ are: (1) to be shared across different views in order to be able to train them only with a small set of real images, and (2) to learn the importance of each part. For example, some parts are occluded by other objects more often than others, and some parts are more discriminative than others. The importance weight $\alpha_i$ will encode this information.

structed scene information could benefit our algorithm to either prune the search space or to assign a prior, and vice versa.

The recent availability of affordable depth sensors has also led to an increase in the use of RGB-D data [21, 11, 3, 30, 26] to extract various types of 3D information from scenes such as depth, object pose, and intrinsic images. Our method can be easily modified to apply to this domain. For example, we can use depth as a cue to improve the quality of the local matching score.

In terms of the goal, our paper is most closely related to [27, 2, 20]. All these papers have a common theme of utilizing the synthetic 3D models for object detection and/or pose estimation. For example, all [27, 2, 20] and this paper infer part locations directly from the 3D models. [2] utilizes training part templates from lots of rendered images and applies to the object detection task. This paper also utilizes the rendered views for training part templates, but our work adds an explicit sharing part importance learned together with a small number of real images. Similar to this paper, the goal of [27] is to estimate the fine poses of objects using exact 3D models. However, unlike that work, we use a combination of modern object detection systems [8] and 3D part sharing to achieve a significant improvement in both detection accuracy and pose recall. Our algorithm is also more efficient ($\sim$ 5x speedup) despite searching over a larger space of potential poses.

In terms of the algorithm, our paper is most similar to [31, 10]. They extend the deformable part model [8] to perform 3D pose detection. However, [10] requires the use of real images as training data, which can be extremely costly to collect for the task of fine pose estimation given the large number of views required ($\sim$ 324k/object). In contrast, our work only requires a small set of real

images for finalizing our model. On the other hand, similar to our work, [31] uses 3D CAD models. However, unlike our work [31] uses rendered images without adapting to the statistics of real images.

## 3 FPM: Fine pose Parts-based Model

Given a set of CAD models, our goal is to accurately detect and pose-align them in RGB images if they contain instances of those objects. Let us formulate this problem in the standard object detection framework, say DPM [8]: we would (1) divide a single object into multiple mixture components - one corresponding to each 3D pose of the object we want to detect, (2) perform detection and (3) attempt to determine the exact mixture component that is responsible for the detected object so that we could identify the pose of the object. Let us do a simple thought experiment to determine the number of mixtures required: first, to parameterize the pose of an object, we require 6 values (3 rotation and 3 translation parameters). Then, we perform a rather coarse discretization of these parameters: say 5 $x$- and $y$-translations, 5 depths, 30 $y$-rotations, 5$x$- and $z$-rotations. This leads to 93750 mixture components[1] compared to the 2 or 3 usually used in DPM!

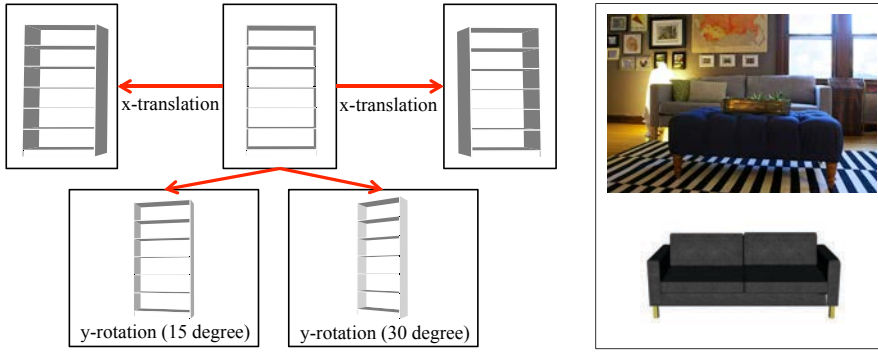
Now, let us assume that it were possible to do inference efficiently with 94k mixtures (each containing multiple parts!), but what about training these models? Even if we used a single example for training each mixture, we would need to take 94k images of a single object. This can be extremely difficult, if not impossible, to capture using physical objects.

In this work, we use CAD models to address the drawbacks above to efficiently and accurately detect and estimate the fine pose of objects using a small set of real images. We summarize the **advantages** of using CAD models below:

- CAD models allow us to render virtually an infinite set of views and discern between them relative to a given point of reference.
- Since we have exact CAD models of objects, we do not need to allow significant deformation between parts. Thus, given the pose, we can estimate the exact location of parts in the image.
- With a CAD model, parts are defined in 3D space allowing us to share information of each part across views. This allows us to use a limited set of training data to learn the importance of each part in 3D, which can then generalize to novel views.

While CAD models provide an appealing source of data, they have one major **disadvantage**: they can be difficult to combine with real images, as the statistics of rendered and real images are often significantly different. For example, as shown in Figure 3(b), occlusions by other objects do not usually occur in rendered images, and further, the appearance of the two domains is also significantly different e.g. the CAD model does not have well textured surfaces.

[1] In our implementation we actually have 324,000 components.




(a) Different shapes from translation and rotation (b) Difference of modalities

**Fig. 3. (a)** This figure illustrates different shapes from various poses. It shows that fine poses cannot be covered by simply rotating objects at the center of the image. For example, $y$-rotation yields a different shape from $x$-translation. The line directions of the shelves do not match due to perspective projection. **(b)** Comparison between a real image and a rendered image. Rendered images do not have occlusion by other objects and lack other appearance details often seen in real images.

Thus, we need a model that combines the advantages of CAD models as described above, while addressing the difference in modality. To achieve this, we propose the Fine pose Parts-based Model (FPM) that combines the deformable part model (DPM)[8] trained on rendered images with objectness scores measured on real images. Specifically, we extend the DPM to have 3D shared parts that are trained using a large number of rendered images and a small number of real images. While the DPM with shared parts allows us to accurately estimate the pose of the object, the objectness scores allow us to leverage an unlabeled set of images to better estimate the location of the objects in the image. We describe our model in Section 3.1, the learning algorithm in Section 3.2 and finally the method of inference in Section 3.3.

### 3.1 Model

Here we define the problem more precisely: given a set of CAD models, and a small set of real images with the pose of the corresponding CAD models present in it, we want to train a model to perform fine pose estimation on new images containing the same set of objects, but with potentially different poses. Thus, we need a model that can robustly identify poses in test images that were not present in the training images.

In order to do this, we define a function $F_\Theta(x)$ that determines how well the pose, $\Theta$, of a CAD model fits a rectangular image window, $x$:

$$F_\Theta(x) = \boldsymbol{\alpha}^\top \mathbf{S}_\Theta(x) + \boldsymbol{\beta}^\top \mathbf{O}_\Theta(x) + \boldsymbol{\gamma}^\top \mathbf{Q}_\Theta \tag{1}$$

where $\mathbf{S}_\Theta$, $\mathbf{O}_\Theta$ and $\mathbf{Q}_\Theta$ are the scores from the DPM with shared parts, objectness and model quality, respectively. Further, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are the parameters

defining the relative weight between the three terms learned using a max-margin framework as described in Section 3.2. Our goal is to maximize $F_\Theta$ for positive images/poses while minimizing it for negative images/poses. We describe the three terms, $\mathbf{S}_\Theta$, $\mathbf{O}_\Theta$ and $\mathbf{Q}_\Theta$ in detail below:

**$\mathbf{S}_\Theta(x)$ - DPM with 3D shared parts:** In this section, we describe how to extend the deformable parts based model (DPM) [8] to work in our scenario. This is a challenging problem because we do not have real images for all possible poses of the objects that we want to be able to identify, but have CAD models instead with a small set of pose-aligned images. Here, we extend the standard formulation of DPM to better exploit the extra information available through the use of CAD models, such as self-occlusion. Further, we show how to propagate information from a small set of pose-aligned images to detect novel poses in images through the use of 3D shared parts.

First, we begin by describing a simple parts-based model that can be trained using rendered images alone. For an image window $x$, and a pose $\Theta$ of a particular CAD model, the score, $s_\Theta$, is given by:

$$s_\Theta(x) = \max_{P_i} \left[ s^r_\Theta(x) + \sum_i s^p_\Theta(P_i, x) \right] \tag{2}$$

where $s^r_\Theta(x) = \mathbf{w}_\Theta \cdot x_{HOG}$ is the score of the root template with pose $\Theta$, and $s^p_\Theta(P_i, x) = \mathbf{w}^{P_i}_\Theta \cdot x_{HOG} - \phi \cdot d_i$ is the score of part template $i$ with location $P_i$ relative to the window $x$. $x_{HOG}$ refers to the HOG [5] features extracted from the image window $x$, and $\phi$ refers to the deformation cost of part $i$ being at a distance $d_i$ from its expected location in the image. We can consider each discretized $\Theta$ as a traditional mixture (i.e. 324,000 mixtures in our case). $\mathbf{w}_\Theta$ refers to the weight vector for the root template and $\mathbf{w}^{P_i}_\Theta$ refers to weight vector for part template $i$ at pose $\Theta$. Since exact 3D models are used here, we do not expect significant deformation of parts relative to the root template given a particular pose. Thus, we manually fix $\phi$ to a specific value.

Now, we describe the modifications made to the above model for our setting:

**Obtaining parts:** Unlike [8], we do not treat parts as latent variables. Instead, we find parts in 3D space by identifying 'joints' in the 3D model as shown in Figure 4(a). Further, when adjacent joints connected via the mesh exceed a certain distance, an additional part is added in between. This results in about 10 to 30 parts per CAD model.

**Learning mixture components:** Given the large number of mixture components, it can be intractable to learn the weights $\mathbf{w}_\Theta$ using an SVM even when using rendered images because of computationally expensive steps such as hard-negative mining. As described in [17], we can use exemplar-LDA to efficiently learn the weights for the root and part templates. Thus, for a given CAD model, the weights, $\mathbf{w}_\Theta$, can be approximated as:

$$\hat{\mathbf{w}}_\Theta = \Sigma^{-1}_{real}(\mu^+_\Theta - \mu^-_{real}) \tag{3}$$

where $\mu^{+}_{\Theta}$ refers to the mean of the HOG features of images rendered at pose $\Theta$, while $\Sigma_{real}$ and $\mu_{real}$ are the covariance matrix and mean vector computed from *real* images. It is important to learn $\Sigma_{real}$ and $\mu_{real}$ from real images in order to account for the difference in modalities. Note that we do not require annotated images to learn these matrices[2], and they can be efficiently modified for templates with different sizes as described in [17]. We follow a similar procedure to learn the weights for the parts.

**Part visibility:** Since we use 3D CAD models, we can easily determine when a part becomes invisible due to self-occlusion. Thus, we multiply the $s^{p}_{\Theta}$ term in Eq. 2 with a binary variable $v_{\Theta}(P_i)$ for the visibility of each individual part $P_i$ at pose $\Theta$. It is set to 1 if a part is visible and 0 otherwise.

**3D shared parts:** As mentioned above, the train data contains a small set of real images with aligned poses - specifically, while we want to distinguish between $324,000$ unique poses, we only have at most 50 real images with aligned poses for training per CAD model in our dataset. How can we use this rather restricted amount of data to learn a model that generalizes to all poses?

Here, we propose to use parts shared in 3D space to propagate the information from the observed poses to the unobserved poses i.e., if we marginalize out the viewpoint, we only need to learn the importance of each 3D part of the model. This allows us to leverage information such as occlusion patterns, and the discriminativeness of different parts from real images. For example, in Figure 3(b), we would expect the sitting area of a sofa to be occluded by other objects (e.g. cushion) more often than the handles of the sofa. Further, some parts (e.g. corners) of a sofa could be more discriminative than others (e.g. lines). Thus, using 3D shared parts, we can propagate this information to all views.

Specifically, we define importance weights, $\alpha_i$ for each 3D shared part. Using 3D CAD models, obtaining part locations and correspondences between two views is trivial. Using the correspondence, we only have to enforce the same $\alpha_i$ for each $P_i$ for all $\Theta$. Figure 4(b) illustrates some of the learned $\alpha_i$ for two different models. We observe that parts that are typically occluded by other objects tend to have lower weights, while parts with more discriminative shapes tend to have higher weights. Similar to [2], $\alpha_i$ is used to rescale the part scores, but in this case we enforce $\alpha_i$ to be shared across views, and learn it directly from real images. Further, it is worth noting that learning $\alpha_i$ would be infeasible in our scenario without sharing.

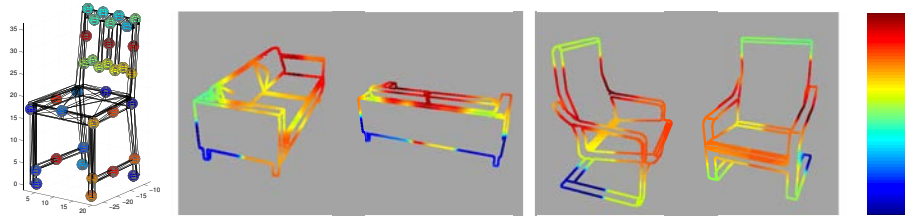
Thus, the term $\mathbf{S}_{\Theta}(x)$ is given by:

$$\mathbf{S}_{\Theta}(x) = \begin{bmatrix} s^{r}_{\Theta}(x) & v_{\Theta}(P_1)s^{p}_{\Theta}(P_1, x) & \dots & v_{\Theta}(P_N)s^{p}_{\Theta}(P_N, x) \end{bmatrix}^{\top} \tag{4}$$

where $N$ is the number of parts in our model, and thus $\boldsymbol{\alpha} \in \mathbb{R}^{N+1}$. Note that $\boldsymbol{\alpha}$ here is the same as that mentioned in Eq. 1. We learn its value jointly with $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ in a max-margin framework, as described in Section 3.2.

**$\mathbf{O}_{\Theta}(x)$ - objectness:** While the first term, $\mathbf{S}_{\Theta}$, in Eq. 1 largely deals with variation in geometry (and some appearance) across views, it suffers from the

[2] We use the covariance matrix and image mean provided by the authors of [17].



(a) Object parts (b) Part importance (increases from blue to red)

**Fig. 4. (a)** Example of object parts for the given model (indicated by the different colored spheres) **(b)** Here, we visualize the importance weights, $\alpha_i$, for each of the parts $i \in \{1 \dots N\}$ as described in Section 3.1. Each part has its own shape template similar to other part-based methods [8]. However, our parts have additional (importance) weights which encode the importance of particular parts. The sharing of these weights enables us to train with rendered images using only a small number of annotated real images. The goal is to learn which part is frequently occluded by other objects or does not contain discriminative shapes from the real data, as this information cannot be derived from 3D CAD models alone. For example, the rear bottom parts of sofa and chair are often occluded by other objects and hence have very low weights.

fact that rendered images are used for training, while real images are used for testing. While we have a limited set of pose-aligned images, we can use the generic concept of objectness [1] to identify whether an image window contains an object or not. We can simply obtain the objectness score for each image window, $x$, by using a typical objectness classifier [1]. However, in order to leverage the representational ability of the recent state-of-the-art deep learning features [25], we simply re-train the objectness classifier (Linear SVM) using selective search [35] and deep learning features extracted using Decaf [6] on the PASCAL VOC dataset [7]. For efficiency, we cache the objectness scores of all the selective search windows in both the train and test splits of our data.

Note that since we use selective search here, we cannot find the objectness scores for the exact windows used in our fine pose detection framework. Instead, given an arbitrary image window $x$, we find the selective search window that has the highest value of intersection over union and use its objectness score. In this way, we can find the value of function $\mathbf{O}_\Theta(x)$ for any image window $x$.

**$\mathbf{Q}_\Theta$ - pose quality:** When training with rendered images, the typical confident false-positives are from the views that are too *general* e.g., the back and side views of a bookcase are simply rectangles; they are not very discriminative and can easily match to a door, a whiteboard, or even a TV screen. We observe that the more near-empty cells a view of a model has, the more it suffers from false-positives. To address this, we use two terms that are suggestive of the *emptiness* of a given model view: (1) the norm of the root weight template at pose $\Theta$, $||\hat{\mathbf{w}}_\Theta||$, and (2) the number of visible mesh surfaces at pose $\Theta$, $n_\Theta$. Thus, $\mathbf{Q}_\Theta = [||\hat{\mathbf{w}}_\Theta||, \ \ n_\Theta]^\top$.

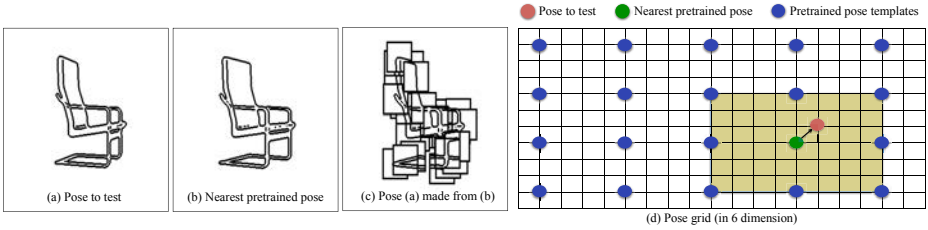



**Fig. 5. Pose sliding window:** During inference, when we match pose $\Theta$ (red dot) to the given image, we use the part templates from the nearest pre-trained pose $\Theta'$ (green dot). The only change from $\Theta'$ is the part location. Part locations are estimated based on the target pose $\Theta$, and the part templates are trained based on $\Theta'$. Blue dots indicate the poses that are pre-trained and hence have part templates, and yellow region indicates the area where the green dot could be covering.

### 3.2 Learning

In this section, we describe how we learn the parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ of the model defined in Eq. 1. Note that all the functions $\mathbf{S}_\Theta$, $\mathbf{O}_\Theta$ and $\mathbf{Q}_\Theta$ as described in Section 3.1 are well defined i.e. they do not contain parameters that we need to learn here. Thus, Eq, 1 becomes a linear system that we can solve in a max-margin framework. Specifically, we learn $W = [\boldsymbol{\alpha} \;\; \boldsymbol{\beta} \;\; \boldsymbol{\gamma}]^\top$ using a linear SVM where the positive pose-aligned images are given in the training set, and we obtain negatives using random sampling of pose across different images. Further, we refine the weights using hard negative mining, similar to [8]. The SVM hyperparameter, $C$, is found using 5 fold cross-validation. We apply the same procedure to all CAD models independently.

### 3.3 Inference

During inference, we evaluate the function $F$ as defined in Eq. 1 for each pose $\Theta$. Given the difficulty of training separate classifiers for a continous pose space $\Theta$, we instead discretize the pose into 9 $x$- and $y$-translations, 5 depths, 32 $y$-rotations, 5$x$- and $z$-rotations leading to 324,000 discretized poses $\{\Theta'_i\}$. Then, during inference, for each fine pose $\Theta$, we find the nearest neighbor $\Theta'_i$ and borrow its trained weights for the root ($\hat{\mathbf{w}}_\Theta$) and part templates ($\hat{\mathbf{w}}_\Theta^{(i)}$). Figure 5 illustrates our approach. In essence, this is extremely similar to the sliding window approach, where each model from a discretized view is only run within the neighboring region of pose space where no discretized poses are present.

After all high scoring pose candidates are obtained, we also perform non-maximal suppression in pose space to obtain the final detection results. Further, to make the sliding window computation efficient, we use sparse coding on the weights of the root and part models as described in [14] allowing us to search this rather large space of fine poses in a reasonable amount of time.

## 4 Experiments

In this section, we evaluate the performance of our approach in detail, focusing on its ability to recover fine-pose. In particular, we evaluate both the average precision of pose estimation (Sec 4.1) and recall among top predictions (Sec 4.2). Further, we compare our method on a standard bounding box detection problem to other state-of-the-art methods (Sec 4.3).

**Dataset:** We use the dataset provided in [27] that contains pose-aligned 3D CAD models with real images. Specifically, we use the harder of the two datasets provided in [27]: the IKEA Room dataset. This dataset contains about 500 images where object poses are annotated for all available 3D CAD models.

### 4.1 Fine Pose Estimation

We compare the previous state-of-the-art method [27] and our method on various settings. We used average precision (AP) measure as suggested by [27] to evaluate our performance. Note that this AP measure takes pose into account, and not only the bounding box location. The normalized 3D space distance between the ground truth pose and the predicted pose is thresholded to determine true or false prediction. The results are summarized in Table 1.

We first evaluate our method at various settings: *only root*, *root+part without* $\alpha$, *root+part with* $\alpha$, and *all*. In case of *only root* and *root+part without* $\alpha$, the detectors suffer from many false positives as compared to [27]. Many false positives occur by some poses having abnomarlly higher confidences than others. However, when we add sharing $\alpha$ and the full set of terms as proposed in Eq 1, our performance increases significantly as we leverage information learned from real images (part importance and objectness).

We further analyze how well our method performs at estimating pose when the bounding box localization is correct. For each ground truth object, we find its first estimated pose that has the highest confidence and has at least 0.7 intersection over union with the ground truth. Then, we compute the ratio of the number of correctly pose estimated objects to the total number of objects. Table 2 shows the result. The average performance of pose estimation given the ground truth bounding box is quite high, 0.84. This indicates that the pose alignment is reliable when the method can localize the target object.

Finally, we show qualitative results in Figure 6 and Figure 7. Figure 6 shows the top 4 false positive results with highest confidence from each class. There are interesting common behaviors. Many misalignments are due to shift in the global position but most of major parts are well-aligned. For example, most bookcase false positives are just one shelf unit shifted ground truth. This repetitive structure of bookcases is a common source of false positives and causes the major performance drop. Also, table and chair have most parts well-aligned to the ground truth, but are often assigned the wrong scale or produce flipped orientations. The major performance drop on the sofa class is due to the fact sofas undergo heavy occlusion, have no strong boundaries, and have complex textures compared to other classes. These differences make training with rendered images

harder. Figure 7 shows some of the top correct detections. It is clear that when the correct pose is found, the alignment is reliable.

| | bookcase *billy1* | chair *poang* | bookcase *expedit* | table *lack2* | sofa *karlstad* | bookcase *billy5* | chair *stefan* | sofa *ektorp* | mean |
|---|---|---|---|---|---|---|---|---|---|
| IKEA [27] | 7.53 | 9.43 | 3.28 | 9.14 | 3.22 | 4.21 | 14.87 | 0.48 | 6.52 |
| R | 1.45 | 1.90 | 0.24 | 5.86 | 0.19 | 1.19 | 4.10 | 0.00 | 1.87 |
| R+P | 3.11 | 7.24 | 3.21 | 13.90 | 2.84 | 4.05 | 7.61 | 0.36 | 5.29 |
| R+P+S | 6.37 | 9.11 | 6.78 | 14.00 | 6.23 | 5.34 | 9.66 | 1.80 | 7.41 |
| Full FPM | **10.52** | **14.02** | **9.30** | **15.32** | **7.15** | **6.10** | **16.00** | **5.66** | **10.51** |

**Table 1. Pose AP score:** we compare our method against the previous state-of-the-art algorithm on IKEA room dataset for fine pose estimation. R, P and S refer to Root, Parts and Sharing respectively. When we use Root and/or Parts that are trained only on rendered images, the performance suffers. However, when combining with part sharing and objectness, our performance improves significantly and outperforms the previous state-of-the-art IKEA algorithm by Lim et al [27].

| bookcase *billy1* | chair *poang* | bookcase *expedit* | table *lack2* | sofa *karlstad* | bookcase *billy5* | chair *stefan* | sofa *ektorp* | mean |
|---|---|---|---|---|---|---|---|---|
| 0.85 | 0.93 | 0.96 | 0.90 | 0.82 | 0.75 | 0.90 | 0.63 | 0.84 |

**Table 2. Pose estimation given the oracle:** Here we investigate how well our pose estimation method works when the object detection step was successful. For each ground truth object, we evaluate pose estimation only within the top detected bounding box that has at least 0.7 intersection over union with ground truth. The numbers indicate the proportion of pose estimates, within this set, that are correct. The average performance of pose estimation given the ground truth is quite high, 0.84.

### 4.2 Pose Proposal

Recently, bounding box proposal algorithms have received a lot of attention [35, 1, 13]. The main motivation is to prune search windows by proposing a set of bounding boxes with a high recall on each image that can later be processed using an algorithm that is computationally more expensive. Following this motivation, we evaluate our algorithm for *pose proposal*. Instead of proposing a set of possible good bounding boxes, our algorithm can generate a set of pose candidates. The algorithm stays the same; however the emphasis of evaluation is now focused more on recall than precision, as compared to the AP measure. For example, with the AP measure, if the top 20% predictions are all correct without any further recall, one can still achieve 0.20 AP score; while this would not be able to serve the role of *pose proposal*.

Table 3 shows the result of recall for all methods. Because all methods will have a full recall in the limit, we limit the number of predictions to be under 2,000. Our method shows quite reliable recalls at all classes, while [27] fails severely to recall some classes. We believe this is because our model can handle

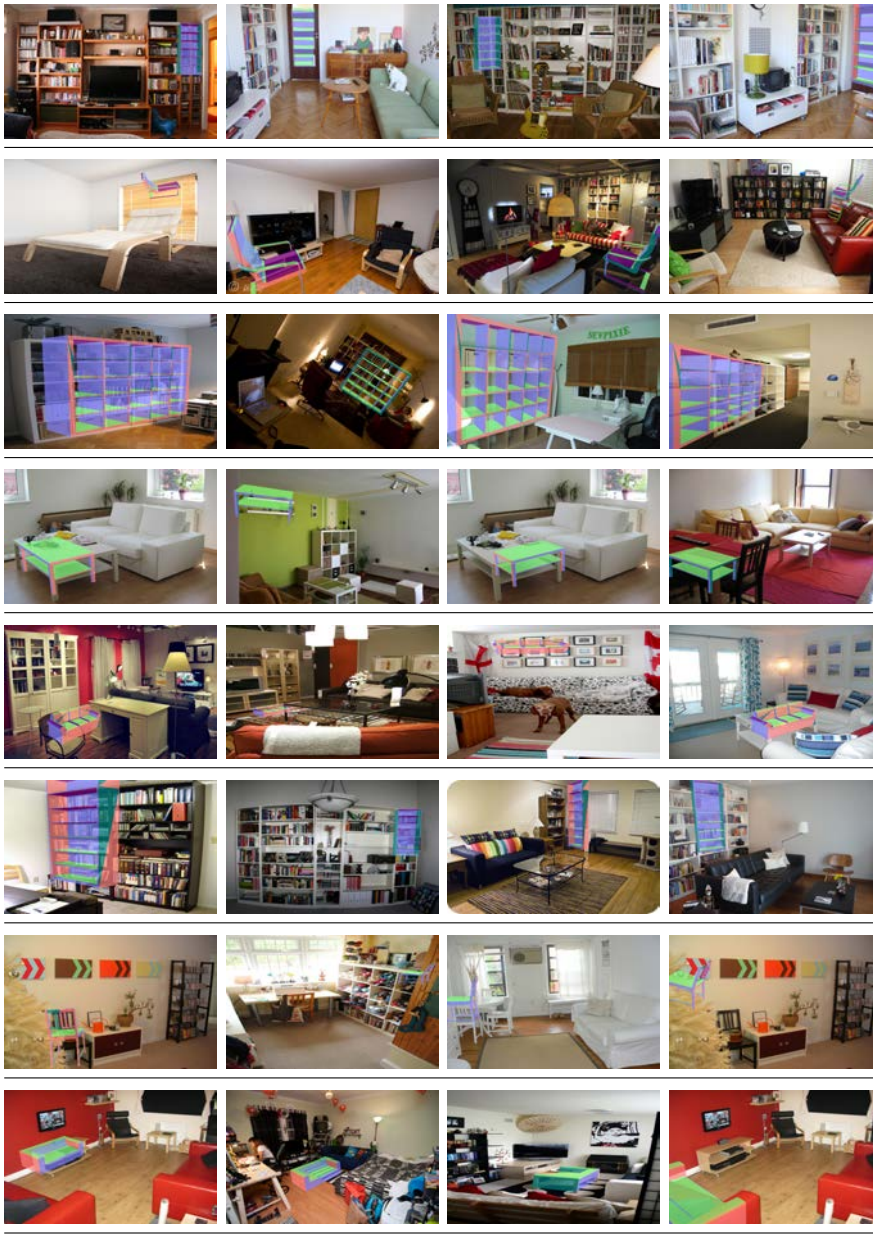



**Fig. 6. Top 4 false positives per class:** (1) Many bookcase false positives are one shelf unit shifted from the ground truth. The repetitive structure of bookcase is a common source of false positives. (2) The major performance drop on the sofa class is due to the fact sofas undergo heavy occlusion, have no strong boundaries, and have complex textures compared to other classes. These difficulties make training with rendered images harder.

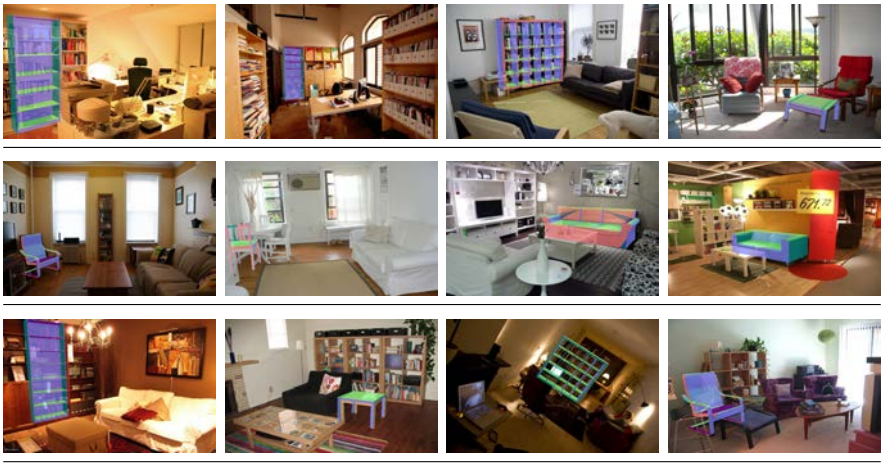


**Fig. 7. Top correct detections:** We show some of the top detections from each class.

| | bookcase *billy1* | chair *poang* | bookcase *expedit* | table *lack2* | sofa *karlstad* | bookcase *billy5* | chair *stefan* | sofa *ektorp* | mean |
|---|---|---|---|---|---|---|---|---|---|
| [27] | 0.83 | 0.86 | 0.89 | 0.83 | 0.37 | 0.88 | 0.77 | 0.56 | 0.75 |
| FPM | **0.87** | **0.93** | **0.94** | **0.88** | **0.83** | **0.91** | **0.99** | **0.91** | **0.91** |

**Table 3. Pose Proposal:** we measure the recall of our method among the top 2000 windows. It shows that our method can recover most of the poses for all objects within the top 2000 predictions. We also outperform [27].

flexible views without being limited to the preset variations of views. [27] depends on a bounded number of views and uses a RANSAC process that can fall into local minima. On the other hand, our method exhaustively examines all possible views. From this result, we can conclude that our method can effectively propose candidate poses for other post-processing algorithms (e.g. context modeling or segmentation).

### 4.3 Bounding Box Detection

While object bounding box detection is not our main target problem, our approach can nonetheless be used for this purpose. For pose estimation algorithms, we extract bounding boxes from predicted poses and apply non-max suppression from [8] before evaluation. We however train [8] with real images using the author-provided code. We evaluate the results at two different thresholds on bounding box intersection over union to examine an ability to capture fine details.

At a lower threshold of 0.5, DPM [8] performs strongly, and our method is relatively weak (in Table 4a). However, at a higher threshold of 0.8, our method

| | bookcase *billy1* | chair *poang* | bookcase *expedit* | table *lack2* | sofa *karlstad* | bookcase *billy5* | chair *stefan* | sofa *ektorp* | mean |
|---|---|---|---|---|---|---|---|---|---|
| (a) Intersection over Union $\geq 0.5$ | | | | | | | | | |
| IKEA [27] | 24.41 | 28.32 | 21.73 | 11.12 | 22.65 | 11.22 | 28.57 | 2.37 | 18.80 |
| DPM [8] | **49.89** | **51.63** | **71.87** | **48.85** | 34.01 | **42.11** | **45.34** | **28.80** | **46.56** |
| FPM | 23.51 | 29.83 | 37.26 | 38.16 | **35.85** | 33.00 | 30.52 | 27.13 | 31.91 |
| (b) Intersection over Union $\geq 0.8$ | | | | | | | | | |
| IKEA [27] | **20.34** | 14.43 | 15.74 | 9.14 | 15.32 | 7.73 | 20.45 | 1.58 | 13.09 |
| DPM [8] | 9.41 | 15.58 | 15.47 | 10.02 | 20.12 | 3.05 | 20.44 | 11.59 | 13.21 |
| FPM | 17.37 | **22.36** | **22.89** | **29.88** | **22.26** | **8.71** | **24.31** | **12.64** | **20.05** |

**Table 4. Bounding box AP measure:** we compare our method against other state-of-the-art algorithms on the IKEA room dataset for bounding box detection. As we increase the threshold to 0.8, the performance of DPM drops significantly; while our method maintains the performance. It shows that our method is capable for fine detection. Our method also significantly outperforms the previous state-of-the-art algorithm [27] in both scenarios, obtaining higher AP for most of the object categories.

outperforms [8] (in Table 4b). This result shows that [8] has a good coarse localization ability but fails capturing fine details. Because our method learns based on each fine pose, the result tends to have a better overlap with the ground truth than that of DPM. Note that DPM is fully trained with real images, and our method is trained based on rendered images and used real images to adapt. Further, we could improve our performance at the 0.5 detection threshold by incorporating scores from DPM into our model.

## 5 Conclusion

In this paper, we introduced a novel approach for fine-pose estimation that leverages geometric information from CAD models to address the problem of fine pose estimation. We show that our method is successfully able to combine the appearance information from relatively few real training images with rendered images from CAD models, outperforming existing approaches in various tasks. Notably, our algorithm significantly outperforms deformable part-based models [8] for high overlap detection, and significantly improves pose recall as compared to [27]. We believe that our work provides a platform to tackle higher level tasks such as contextual reasoning in scenes using the fine poses of objects.

**Acknowledgements**

We thank Hamed Pirsiavash for inspiring discussions, and Phillip Isola, Andrew Owens, and Carl Vondrick for many helpful comments.

## References


1. Alexe, B., Deselaers, T., Ferrari, V.: What is an object? In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 73–80 (2010)
2. Aubry, M., Maturana, D., Efros, A., Russell, B., Sivic, J.: Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In: IEEE Conference on Computer Vision and Pattern Recognition (2014)
3. Barron, J.T., Malik, J.: Intrinsic scene properties from a single rgb-d image. IEEE Conference on Computer Vision and Pattern Recognition (2013)
4. Choi, W., Chao, Y.W., Pantofaru, C., Savarese, S.: Understanding indoor scenes using 3d geometric phrases. In: IEEE Conference on Computer Vision and Pattern Recognition (2013)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition (2005)
6. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. arXiv preprint arXiv:1310.1531 (2013)
7. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results
8. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.: Discriminatively trained deformable part models (2009)
9. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. International Journal of Computer Vision 61(1), 55–79 (2005)
10. Fidler, S., Dickinson, S.J., Urtasun, R.: 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In: Advances in Neural Information Processing Systems (2012)
11. Fisher, M., Hanrahan, P.: Context-based search for 3d models. ACM Trans. Graph. 29(6) (Dec 2010)
12. Fouhey, D.F., Delaitre, V., Gupta, A., Efros, A.A., Laptev, I., Sivic, J.: People watching: Human actions as a cue for single-view geometry. In: European Conference on Computer Vision (2012)
13. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (2014)
14. Girshick, R., Song, H.O., Darrell, T.: Discriminatively activated sparselets. In: International Conference on Machine Learning (2013)
15. Gupta, A., Satkin, S., Efros, A.A., Hebert, M.: From 3d scene geometry to human workspace. In: IEEE Conference on Computer Vision and Pattern Recognition (2011)
16. Gupta, S., Arbelaez, P., Malik, J.: Perceptual organization and recognition of indoor scenes from RGB-D images. IEEE Conference on Computer Vision and Pattern Recognition (2013)
17. Hariharan, B., Malik, J., Ramanan, D.: Discriminative decorrelation for clustering and classification. In: European Conference on Computer Vision (2012)
18. Hedau, V., Hoiem, D., Forsyth, D.: Thinking inside the box: using appearance models and context based on room geometry. In: European Conference on Computer Vision (2010)
19. Hejrati, M., Ramanan, D.: Analyzing 3d objects in cluttered images. In: Advances in Neural Information Processing Systems (2012)

20. Hejrati, M., Ramanan, D.: Analysis by synthesis: 3d object recognition by object reconstruction. In: IEEE Conference on Computer Vision and Pattern Recognition (2014)
21. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: Rgbd mapping: Using depth cameras for dense 3d modeling of indoor environments. In: RGB-D: Advanced Reasoning with Depth Cameras Workshop in conjunction with RSS (2010)
22. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: IEEE International Conference on Computer Vision (2005)
23. Hoiem, D., Hedau, V., Forsyth, D.: Recovering free space of indoor scenes from a single image. In: IEEE Conference on Computer Vision and Pattern Recognition (2012)
24. Jia, Z., Gallagher, A., Saxena, A., Chen, T.: 3d-based reasoning with blocks, support, and stability. In: IEEE Conference on Computer Vision and Pattern Recognition (2013)
25. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (2012)
26. Lai, K., Bo, L., Ren, X., Fox, D.: Detection-based object labeling in 3d scenes. In: IEEE International Conference on on Robotics and Automation (2012)
27. Lim, J.J., Pirsiavash, H., Torralba, A.: Parsing ikea objects: Fine pose estimation. In: IEEE International Conference on Computer Vision (2013)
28. Lowe, D.: Fitting parameterized three-dimensional models to images. IEEE Transactions on Pattern Analysis and Machine intelligence (1991)
29. Matzen, K., Snavely, N.: Nyc3dcars: A dataset of 3d vehicles in geographic context. In: Proc. Int. Conf. on Computer Vision (2013)
30. Nathan Silberman, Derek Hoiem, P.K., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: European Conference on Computer Vision (2012)
31. Pepik, B., Gehler, P., Stark, M., Schiele, B.: 3d2pm - 3d deformable part models. In: European Conference on Computer Vision (2012)
32. Satkin, S., Lin, J., Hebert, M.: Data-driven scene understanding from 3D models. In: British Machine Vision Conference (2012)
33. Schwing, A.G., Fidler, S., Pollefeys, M., Urtasun, R.: Box In the Box: Joint 3D Layout and Object Reasoning from Single Images. In: Proc. ICCV (2013)
34. Sun, M., Su, H., Savarese, S., Fei-Fei, L.: A multi-view probabilistic model for 3d object classes. In: IEEE Conference on Computer Vision and Pattern Recognition (2009)
35. Uijlings, J., van de Sande, K., Gevers, T., Smeulders, A.: Selective search for object recognition. International Journal of Computer Vision (2013)
36. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond pascal: A benchmark for 3d object detection in the wild. In: IEEE Winter Conference on Applications of Computer Vision (2014)
37. Xiao, J., Russell, B., Torralba, A.: Localizing 3d cuboids in single-view images. In: Advances in Neural Information Processing Systems (2012)
38. Zhao, Y., Zhu, S.C.: Scene parsing by integrating function, geometry and appearance models. In: IEEE Conference on Computer Vision and Pattern Recognition (2013)
39. Zia, M., Stark, M., Schindler, K.: Explicit occlusion modeling for 3d object class representations. In: IEEE Conference on Computer Vision and Pattern Recognition (2013)